IMPROVED TRAINING OF CERTIFIABLY ROBUST MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Convex relaxations are effective for training and certifying neural networks against norm-bounded adversarial attacks, but they leave a large gap between certifiable and empirical (PGD) robustness. In principle, relaxation can provide tight bounds if the convex relaxation solution is feasible for the original non-relaxed problem. Therefore, we propose two regularizers that can be used to train neural networks that yield convex relaxations with tighter bounds. In all of our experiments, the proposed regularizations result in tighter certification bounds than non-regularized baselines.

1 INTRODUCTION

Neural networks achieve excellent performances on many computer vision tasks, but they could be vulnerable to small, adversarially chosen perturbations that are barely perceptible to humans while having a catastrophic impact on the network's performance (Szegedy et al., 2013; Goodfellow et al., 2014). Making classifiers robust to these adversarial perturbations is of great interest, especially when neural networks are applied to safety-critical applications. Several heuristic methods exist for obtaining robust classifiers, however powerful adversarial examples can be found against most of these defenses (Carlini & Wagner, 2017; Uesato et al., 2018).

Recent studies focus on verifying or enforcing the certified accuracy of deep classifiers, especially for networks with ReLU activations. They provide guarantees of a network's robustness to any perturbation δ with norm bounded by $\|\delta\|_p \leq \epsilon$ (Wong & Kolter, 2017; Wong et al., 2018; Raghunathan et al., 2018; Dvijotham et al., 2018b; Zhang et al., 2018; Salman et al., 2019). There are exact verifiers that find the exact minimum adversarial distortions δ or the robust error (Ehlers, 2017; Katz et al., 2017; Tjeng et al., 2017), but due to the non-convex nature of the problem, exact verification is NP-hard. To make verification efficient and scalable, convex relaxations are adopted to expand the non-convex feasible set given by non-linear activations into convex sets, resulting in a lower bound on the norm of adversarial perturbations (Zhang et al., 2018; Weng et al., 2018), or an upper bound on the robust error (Dvijotham et al., 2018b; Gehr et al., 2018; Singh et al., 2018). With LP relaxations (Wong et al., 2018), such a verifier can be efficient enough to estimate the lower bound of the margin in each iteration for training certifiably robust networks. However, due to the relaxation of the underlying problem, the lower bound of the margin is potentially loose, and thus a barrier remains between the optimal values from the original and relaxed problems (Salman et al., 2019).

In this paper, we focus on improving the certified robustness of neural networks trained with convex relaxation bounds. To achieve this, we first give a more interpretable explanation for the bounds achieved in (Weng et al., 2018; Wong et al., 2018). Namely, the constraints of the relaxed problem are defined by a simple linear network with adversaries injecting bounded perturbations to both the input of the network and the pre-activations of intermediate layers. The optimal solution of the relaxed problem can be written as a forward pass of the clean image through the linear network, plus the cumulative adversarial effects of all the perturbations added to the linear transforms, which makes it easier to identify the optimality conditions and serves as a bridge between the relaxed problem and the original non-convex problem. We further identify conditions for the bound to be tight, and we propose two indicators for the gap between the original non-convex problem and the relaxed problem. Adding the proposed indicators into the loss function results in classifiers with better certified accuracy.

2 BACKGROUND AND RELATED WORK

Adversarial defenses roughly fall into two categories: heuristic defenses and verifiable defenses. The heuristic defenses either try to identify adversarial examples and remove adversarial perturbations from images, or make the network invariant to small perturbations through training (Papernot & McDaniel, 2018; Shan et al., 2019; Samangouei et al., 2018; Hwang et al., 2019). In addition, adversarial training uses adversarial examples as opposed to clean examples during training, so that the network can learn how to classify adversarial examples directly (Madry et al., 2017; Shafahi et al., 2019; Zhang et al., 2019a).

In response, a line of works have proposed to verify the robustness of neural nets. Exact methods obtain the perturbation δ with minimum $\|\delta\|_p$ such that $f(x) \neq f(x+\delta)$, where f is a classifier and x is the data point. Nevertheless, the problem itself is NP-hard and the methods can hardly scale (Cheng et al., 2017; Lomuscio & Maganti, 2017; Dutta et al., 2018; Fischetti & Jo, 2017; Tjeng et al., 2017; Scheibler et al., 2015; Katz et al., 2017; Carlini et al., 2017; Ehlers, 2017).

A body of work focuses on relaxing the non-linearities in the original problem into linear inequality constraints (Singh et al., 2018; Gehr et al., 2018; Zhang et al., 2018; Mirman et al., 2018), sometimes using the dual of the relaxed problem (Wong & Kolter, 2017; Wong et al., 2018; Dvijotham et al., 2018b). Recently, Salman et al. (2019) unified the primal and dual views into a common convex relaxation framework, and suggested there is an inherent gap between the actual robustness and the lower bound of distortion achieved by verifiers that use LP relaxations, which they called a *convex relaxation barrier*.

Some defense approaches integrate the verification methods into the training of a network to minimize robust loss directly. (Hein & Andriushchenko, 2017) uses a local lipschitz regularization to improve certified robustness. In addition, a bound based on semi-definite programming (SDP) relaxation was developed and minimized as the objective Raghunathan et al. (2018). (Wong & Kolter, 2017) presents an upper bound on the robust loss caused by norm-bounded perturbation via LP relaxation, and minimizes this upper bound during training. In (Wong et al., 2018), they further extend this method to much more general network structures with skip connections and general non-linearities, and provide a memory-friendly training strategy using random projections. Since LP relaxation is adopted, the aforementioned convex relaxation barrier exists for their methods.

While another line of work (IBP) have shown that an intuitively looser interval bound can be used to train much more robust networks than convex relaxation for large ℓ_∞ perturbations (Gowal et al., 2018; Zhang et al., 2019b), it is still important to study convex relaxation bounds since it can certify a broader class of adversaries that IBP cannot certify, like the ℓ_2 adversaries.

We seek to enforce the tightness of the convex relaxation certificate during training. It reduces the optimality gap between the original and the relaxed problem by adding the proposed indicators into the loss function as regularizers. Compared with previous approaches, we have the following contributions: First, based upon the same relaxation in (Weng et al., 2018), we illustrate a more intuitive view for the bounds on intermediate ReLU activations achieved by (Wong et al., 2018) , which can be viewed as a linear network facing adversaries adding bounded perturbations to both the input and the intermediate layers. Second, starting from this view, we identify conditions where the bound from the relaxed problem is tight for the original non-convex problem. Third, based on the conditions, we propose regularizers that encourage the bound to be tight for the obtained network, which improves the certificate on both MNIST and CIFAR-10.

3 PROBLEM FORMULATION

To simplify the notations, without loss of generality, we consider maximizing the lower bound of the margin of a L-layer neural network under norm-bounded perturbations for certifiable defense. In practice, like many related works Wong et al. (2018); Gowal et al. (2018), we use a cross entropy loss on the negation of such margins for classification problems with more than two classes as in Eq. 10. We reformulate the inner minimization problem (by the adversary) into solving a constrained optimization to find the minimum margin under norm-bounded perturbation as

$$\underset{z_1 \in \mathcal{B}_{p,\epsilon}(x)}{\text{minimize}}\ c_t^\top x_L, \text{subject to } z_{i+1} = \sigma(x_i), x_i = f_i(z_i), \text{for } i = 1, ..., L, \qquad (\mathcal{O})$$

where $c_t = e_y - e_t$, e_y and e_t are one-hot vectors corresponding to the label y and some other class t, $\sigma(\cdot)$ is the ReLU activation, and f_i is one functional block of the neural network. This can be a linear layer ($f_i(z_i) = W_i z_i + b_i$), or even a residual block. We use $h_i(x) = f_i(\sigma(f_{i-1}(\cdots f_1(x))))$ to denote the ReLU network up to the i-th layer, and the optimal solution to $\mathcal{O}$ as $p^*_{\mathcal{O}}$.

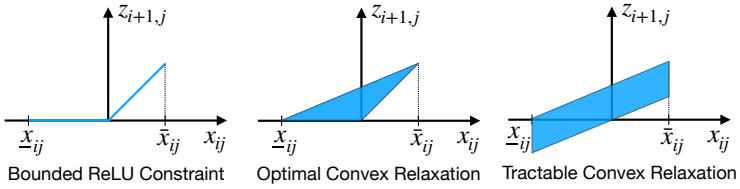

Figure 1: The feasible sets (blue regions/lines) given by the bounded ReLU constraints (Eq. $\mathcal{O}$), convex hull ($\overline{conv}_{ij}$) and the relaxation used in this paper (specific choice for Eq. 2) for $j \in \mathcal{I}_i$.

The constraints of ReLU give rise to a non-convex feasible set, making Eq. $\mathcal{O}$ a non-convex optimization problem. We relax such constraints so that $\mathcal{O}$ becomes a convex optimization problem which lower bounds the objective function in Eq. $\mathcal{O}$. Moreover, we want such a problem to be solved efficiently, or even have a closed form solution. We adopt the same relaxation as introduced in (Weng et al., 2018). Though starting from different points of view, it can be easily proved that the objective derived from a dual view in (Wong et al., 2018) is the same as (Weng et al., 2018). Therefore, our conclusions can be applied to either formulations.

3.1 Interpreting the Relaxation Bound

In this section, we look at a relaxed convex problem of $\mathcal{O}$. For simplicity, here we consider networks with no skip connections. We represent both FC and Conv layers as $f_i(z_i) = W_i z_i + b_i$, where for Conv layers W_i is the equivalent Toeplitz matrix.

Grouping of ReLU Activations Since the network is a continuous function, the pre-activations x_i have lower and upper bounds $\underline{x}_i$ and $\bar{x}_i$ when the input z_1 is bounded. Based on the ranges of $\underline{x}_i$ and $\bar{x}_i$, we divide the ReLU activations into three disjoint subsets

$$\mathcal{I}_i^- = \{j | \bar{x}_{ij} \leq 0\}, \quad \mathcal{I}_i^+ = \{j | \underline{x}_{ij} \geq 0\}, \quad \mathcal{I}_i = \{j | \underline{x}_{ij} < 0 < \bar{x}_{ij}\}. \tag{1}$$

For $\mathcal{I}_i^-$ and $\mathcal{I}_i^+$, the ReLU constraints degenerate to linear constraints, which do not affect convexity. For illustration, if $j \in \mathcal{I}_i^-$, the constraint will be $z_{i+1,j} = 0$, and if $j \in \mathcal{I}_i^+$, the constraint will be $z_{i+1,j} = x_{ij}$. On the other hand, for $\mathcal{I}_i$, the ReLU activations are non-linear. If $j \in \mathcal{I}_i$, we call the corresponding ReLU activation an *unstable neuron*.

Computational Challenge for the "optimal" Relaxation As pointed out by (Salman et al., 2019), the optimal relaxation for each $j \in \mathcal{I}_i$ is the closed convex hull $\overline{conv}_{ij}$ of $\mathcal{S}_{ij} = \{(x_{ij}, z_{i+1,j}) | j \in \mathcal{I}_i, z_{i+1,j} = \max(0, x_{ij}), \underline{x}_{ij} \leq x_{ij} \leq \bar{x}_{ij}\}$, which is just $\overline{conv}_{ij} = \{(x_{ij}, z_{i+1,j}) | \max(0, x_{ij}) \leq z_{i+1,j} \leq \frac{\bar{x}_{ij}}{\bar{x}_{ij} - \underline{x}_{ij}}(x_{ij} - \underline{x}_{ij})\}$, corresponding to the triangle region in the middle of Figure 1. Salman et al. (2019) solved this relaxation directly through LP, but it takes 10,000 CPU cores to parallelize the LP solvers needed to bound the activations of every neuron in a two-hidden-layer MLP with 100 neurons per layer.

An Alternative Relaxation To find the lower bound of equation $\mathcal{O}$ which could be efficiently solved and scale to large networks, we relax each non-convex ReLU constraint into two linear inequality constraints corresponding to the lower and upper bounds of the ReLU activation z_{ij}. Specifically, we consider the following Linear Programming:

$$\underset{z_1 \in \mathcal{B}_{p,\epsilon}(x)}{\text{minimize}} \; c_t^\top x_L, \text{subject to } \underline{a}_{ij} x_{ij} + \underline{b}_{ij} \leq z_{i+1,j} \leq \bar{a}_{ij} x_{ij} + \bar{b}_{ij}, x_i = W_i z_i + b_i, \text{for } i = 1, ..., L, \tag{2}$$

where $\{\underline{a}_{ij}, \underline{b}_{ij}, \bar{a}_{ij}, \bar{b}_{ij}\}$ are chosen such that for $j \in \mathcal{I}_i$, $\underline{a}_{ij} x_{ij} + \underline{b}_{ij} \leq \max(0, x_{ij}) \leq \frac{\bar{x}_{ij}}{\bar{x}_{ij} - \underline{x}_{ij}}(x_{ij} - \underline{x}_{ij}) \leq \bar{a}_{ij} x_{ij} + \bar{b}_{ij}$; for $j \in \mathcal{I}_i^+$, $\underline{a}_{ij} = \bar{a}_{ij} = 1, \bar{a}_{ij} = \bar{b}_{ij} = 0$; for $j \in \mathcal{I}_i^-$, $\underline{a}_{ij} = \bar{a}_{ij} = \underline{b}_{ij} =$

$\overline{b}_{ij} = 0$. The optimal solution p^* of Eq. 2 satisfies $p^* \leq p^*_{\mathcal{O}}$, since the feasible set of $\mathcal{O}$ is a subset of Eq. 2. We adopt the setting of (Weng et al., 2018), where for $j \in \mathcal{I}_i$, $\underline{a}_{ij} = \overline{a}_{ij} = \frac{\overline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}$, $\underline{b}_{ij} = 0$, $\overline{b}_{ij} = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}$, shown as the blue region in the right of Figure 1. To compute the lower and upper bound $\underline{x}_{ij}$ and $\overline{x}_{ij}$ for x_{ij}, we just need to replace the objective of Eq. 2 with $c_{ij}^\top x_i$, where c_{ij} is a one-hot vector with the same number of entries as x_i and the j-th entry being 1 for $\underline{x}_{ij}$ and -1 for $\overline{x}_{ij}$ (plus an extra negation).

Interpretation of the Alternative Relaxation via a Linear Network with Injected Noise Like (Weng et al., 2018; Wong et al., 2018; Salman et al., 2019), such constraints allow each intermediate ReLU activation to reach their upper or lower bounds independently. As a result, each intermediate *unstable neuron* can be seen as an adversary adding a perturbation δ_{ij} in the range $[0, -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}]$ to a linear transform, represented as $z_{ij} = \frac{\overline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}} x_{ij} + \delta_{ij}$. Such a point of view gives rise to a more interpretable explanation for the bounds achieved in (Weng et al., 2018). If we construct a network from the relaxed constraints, then the problem becomes how to choose the perturbations for both the input and intermediate unstable neurons to minimize $c_t^\top x_L$ of a multi-layer linear network. Such a linear network under the perturbations is defined as

$$z_{i+1} = D_i x_i + \delta_i, x_i = W_i z_i + b_i, \text{ for } i = 1, ..., L, z_1 = x + \delta_0, \tag{3}$$

where D_i is a diagonal matrix and δ_i is a vector. The input perturbation satisfies $\|\delta_0\|_p \leq \epsilon$. The j-th diagonal entry D_{ij} and the j-th entry δ_{ij} for $i > 0$ is defined as

$$D_{ij} = \begin{cases} 0, & \text{if } j \in \mathcal{I}_i^- \\ 1, & \text{if } j \in \mathcal{I}_i^+ \\ \frac{\overline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}, & \text{if} j \in \mathcal{I}_i \end{cases}, \delta_{ij} \in \mathcal{S}_{\delta_{ij}} = \begin{cases} \left[0, -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}\right], & \text{if } j \in \mathcal{I}_i \\ \{0\}, & \text{otherwise.} \end{cases} \tag{4}$$

With such an observation, we can further unfold the objective in Eq. 2 into a more interpretable form as

$$c_t^\top x_L = c_t^\top f_L(D_{L-1} f_{L-1}(\cdots D_1 f_1(x))) + c_t^\top \sum_{i=0}^{L-1} W_L \prod_{k=i+1}^{L-1} D_k W_k \delta_i, \tag{5}$$

where the first term of RHS is a forward pass of the clean image x through a linear network interleaving between a linear layer $x = W_i z + b_i$ and a scaling layer $z = D_i x$, and the second term is the sum of the i-th perturbation passing through all the weight matrices W_i of the linear operation layers and scaling layers D_i after it.

Therefore, under such a relaxation, only the second term is affected by the variables $\{\delta_i\}_{i=0}^{L-1}$ for optimizing Eq. 2. Denote the linear network up to the i-th layer as $g_i(x)$, and $\mathcal{W}_{i:i'} = W_i \prod_{k=i'}^{i-1} D_k W_k$. We can transform Eq. 2 with $\underline{a}_{ij} = \overline{a}_{ij} = \frac{\overline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}$, $\underline{b}_{ij} = 0$, $\overline{b}_{ij} = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}$ into the following constrained optimization problem

$$\underset{\delta_0, ..., \delta_{L-1}}{\text{minimize}}\ c_t^\top g_L(x) + \sum_{i=0}^{L-1} c_t^\top \mathcal{W}_{L:i+1} \delta_i, \text{ subject to } \|\delta_0\|_p \leq \epsilon, \delta_{ij} \in \mathcal{S}_{\delta_{ij}} \text{ for } i = 1, ..., L-1. \quad (\mathcal{C})$$

Notice $c_t^\top \mathcal{W}_{L:i+1}$ is just a row vector. For $i > 0, j \in \mathcal{I}_i$, to minimize $c_t^\top \mathcal{W}_{L:i+1} \delta_{ij}$, we let δ_{ij} be its minimum value 0 if $(c_t^\top \mathcal{W}_{L:i+1})_j \geq 0$, or its maximum value $\delta_{ij} = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}$ if $(c_t^\top \mathcal{W}_{L:i+1})_j < 0$, where $(c_t^\top \mathcal{W}_{L:i+1})_j$ is the j-th entry of $c_t^\top \mathcal{W}_{L:i+1}$. For δ_0, when the infinity norm is used, we set $\delta_{ij} = -\epsilon$ if $(c_t^\top \mathcal{W}_{L:i+1})_j \geq 0$, and otherwise $\delta_{ij} = \epsilon$. For other norms, it is also easy to see that

$$\min_{\|\delta_0\|_p \leq \epsilon} c_t^\top \mathcal{W}_{L:1} \delta_0 = -\epsilon \max_{\|\delta_0\|_p \leq 1} -c_t^\top \mathcal{W}_{L:1} \delta_0 = -\epsilon \|c_t^\top \mathcal{W}_{L:1}\|_*, \tag{6}$$

where $\|\cdot\|_*$ is the dual norm of the p norm. In this way, the optimal value $p^*_{\mathcal{C}}$ of the relaxed problem (Eq. $\mathcal{C}$) can be found efficiently without any gradient step. The resulting expression for the lower-bound is

$$p^*_{\mathcal{O}} \geq p^*_{\mathcal{C}} = c_t^\top g_L(x) - \epsilon \|c_t^\top \mathcal{W}_{L:1}\|_* - \sum_{i=1}^{L-1} \sum_{j \in I_i} \frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}} \min((c_t^\top \mathcal{W}_{L:i+1})_j, 0). \tag{7}$$

4 TIGHTER BOUNDS VIA REGULARIZATION

4.1 CONDITIONS FOR EQUALITY

Despite being efficient to compute, Eq. $\mathcal{C}$ is not necessarily the tightest ("optimal") convex relaxation as pointed out by (Salman et al., 2019). We investigate the gap between the optimal convex relaxation in Eq. $\mathcal{O}$ and the alternative computationally efficient convex relaxation in Eq. $\mathcal{C}$ which lower bounds Eq. $\mathcal{O}$. The gap between the two could be 0 if the convex solution in Eq. $\mathcal{C}$ is feasible in Eq. $\mathcal{O}$.

Here we look into conditions that make $p^*_{\mathcal{O}} = p^*_{\mathcal{C}}$. Generally, as long as the optimal solution $\{\delta^*_i\}_{i=0}^{L-1}$ of Eq. $\mathcal{C}$ is "equivalent" to a feasible solution $\{z'_i, x'_i\}_{i=1}^{L}$ of Eq. $\mathcal{O}$, we will have $p^*_{\mathcal{O}} = p^*_{\mathcal{C}}$. Since if $p^*_{\mathcal{C}}$ is feasible for Eq. $\mathcal{O}$, then $p^*_{\mathcal{O}} \leq p^*_{\mathcal{C}}$, meanwhile $p^*_{\mathcal{O}} \geq p^*_{\mathcal{C}}$ due to the relaxation.

Let $\{z'_i, x'_i\}_{i=1}^{L-1}$ denote a set of feasible solutions of $\mathcal{O}$ computed by passing z'_1 through the ReLU sub-networks $h_i(z'_1)$ defined in $\mathcal{O}$. And denote the resulting feasible objective value as $p'_{\mathcal{O}} = c_t^\top x'_L$.

A relaxed and non-relaxed solution to be equivalent (and relaxation will be exact) when two conditions hold. First, the architecture and weights of the networks are the same. Second the input to both networks are z^*_1, and we have $z^*_i = z'_i$ and $x^*_i = x'_i$ for $i = 1, ..., L-1$, where $\{z^*_i, x^*_i\}_{i=1}^{L-1}$ is computed by adding $\{\delta^*_i\}_{i=0}^{L-1}$ into the equivalent linear network defined in Eq. 3. In this way, we know $x^*_L = x'_L$ and the margin $c_t^\top x^*_L = c_t^\top x'_L$, so $p^*_{\mathcal{C}} = p'_{\mathcal{O}} = c_t^\top x'_L$. Therefore, to check whether the equivalence exists, given $\{\delta^*_i\}_{i=0}^{L-1}$, we can find $\{z^*_i, x^*_i\}_{i=1}^{L}$ and check if they satisfy the constraints of $\mathcal{O}$. If so, then we will have $p^*_{\mathcal{O}} = p^*_{\mathcal{C}}$.

4.2 A INTUITIVE (BUT WEAK) INDICATOR OF TIGHTNESS: $p'_{\mathcal{O}} - p^*_{\mathcal{C}}$

The observation above motivates us to consider the quantity

$$d(x, \delta^*_0, W, b) = p'_{\mathcal{O}}(x, \delta^*_0) - p^*_{\mathcal{C}} \tag{8}$$

as an indicator of the difference between $\{z^*_i, x^*_i\}_{i=1}^{L-1}$ and $\{z'_i, x'_i\}_{i=1}^{L-1}$, where $p^*_{\mathcal{C}}$ can be computed from Eq. 7, and $p'_{\mathcal{O}}(x, \delta^*_0) = c_t^\top h_L(x + \delta^*_0)$. δ^*_0 can be computed efficiently from the optimality condition of Eq. 6. For example, when $p = \infty$, the optimal input perturbation δ^*_0 of $\mathcal{C}$ is $\delta^*_0 = -\epsilon \text{sign}(c_t^\top \mathcal{W}_{L:1})$, which corresponds to sending $z'_1 = z^*_1 = x - \epsilon \text{sign}(c_t^\top \mathcal{W}_{L:1})$ through the ReLU network; when $p = 2$, $\delta^*_0 = -\epsilon \frac{c_t^\top \mathcal{W}_{L:1}}{\|c_t^\top \mathcal{W}_{L:1}\|_2}$, which corresponds to sending $z'_1 = z^*_1 = x - \epsilon \frac{c_t^\top \mathcal{W}_{L:1}}{\|c_t^\top \mathcal{W}_{L:1}\|_2}$.

The larger $d(x, \delta^*_0, W, b)$ is, the more relaxed $\mathcal{C}$ is, and the higher $p^*_{\mathcal{O}} - p^*_{\mathcal{C}}$ could be. Therefore, we can regularize the network to minimize $d(x, \delta^*_0, W, b)$ during training and maximize the lower-bound of the margin $p^*_{\mathcal{C}}$, so that we can obtain a network where $p^*_{\mathcal{C}}$ is a better estimate of $p^*_{\mathcal{O}}$ and the robustness is better represented by $p^*_{\mathcal{C}}$. Such an indicator avoids comparing and storing the intermediate variables, and requires only one additional forward pass through the ReLU network. It bears some similarities to knowledge distillation (Hinton et al., 2015), since the lower bound $\mathcal{C}$ learns to give similar output of the corresponding ReLU network.

The tightest indicator should give the minimum gap $p^*_{\mathcal{O}} - p^*_{\mathcal{C}}$, where we need to find the optimal perturbation for $\mathcal{O}$. However, the minimum gap cannot be found in polynomial time, due to the non-convex nature of $\mathcal{O}$. (Weng et al., 2018) also proved that there is no polynomial time algorithm to find the minimum ℓ_1-norm adversarial distortion with $0.99 \ln n$ approximation ratio unless NP=P, a problem equivalent to finding the minimum margin here.

Despite being intuitive and exploiting the power of end-to-end training of neural networks, Eq. 8 does not work well in practice, as we will show in the experiments. Unless $d(x, \delta^*_0, W, b) = 0$, $p^*_{\mathcal{C}}$, $\{z^*_i, x^*_i\}_{i=1}^{L-1}$ may deviate a lot from $\{z'_i, x'_i\}_{i=1}^{L-1}$ and does not correspond to any ReLU network, even if $d(x, \delta^*_0, W, b)$ may seem small. For example, it is possible that $z^*_{ij} < 0$ for a given z^*_1, but a ReLU network will always have $z'_{ij} \geq 0$. In most practical cases, we cannot optimize $d(x, \delta^*_0, W, b)$ down to 0, therefore it does not serve as an effective regularizer.

4.3 A BETTER INDICATOR FOR REGULARIZATION

We find an alternative regularizer more effective at improving verifiable accuracy. The regularizer encourages the feasible solution $\{z'_i, x'_i\}_{i=1}^{L-1}$ of $\mathcal{O}$ to *exactly* match the feasible optimal solution

$\{z_i^*, x_i^*\}_{i=1}^{L-1}$ of $\mathcal{C}$. It is easy to see from the plot on the right of Figure 1 that for $j \in \mathcal{I}_i$, the only 3 feasible optimal solutions for $(x_{ij}^*, z_{i+1,j}^*)$ of $\mathcal{C}$ that are also feasible for $\mathcal{O}$ are $(\underline{x}_{ij}, 0), (\overline{x}_{ij}, \overline{x}_{ij})$ and $(0, 0)$, since the optimal solution of $\mathcal{C}$ can only be on the boundary of the feasible set. The first two cases correspond to $\delta_{ij}^* = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}}$, while the last case corresponds to $\delta_{ij}^* = 0$. Specifically, for $j \in \mathcal{I}_i$, from the optimality conditions of $\mathcal{C}$, δ_{ij}^* can be either 0 or $-\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}}$. When $\delta_{ij}^* = 0$ and $x_{ij}^* = x'_{ij}$, to make $z_{i+1,j}^* = z'_{i+1,j}$, we require $\frac{\overline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}} x'_{ij} = \max(x'_{ij}, 0)$. Since $\frac{\overline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}} < 1$, the only solution is $x'_{ij} = 0$. When $\delta_{ij}^* = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}}$, we requires $\frac{\overline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}} x'_{ij} + -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}} = \max(x'_{ij}, 0)$, so the solution is $x'_{ij} = \underline{x}_{ij}$ or $x'_{ij} = \overline{x}_{ij}$. Indeed, we achieve optimality if x'_{ij} and x_{ij}^* satisfy these conditions only at unstable neurons, i.e., for $j \in \mathcal{I}_i, i = 1, ..., L-1$.

Proposition 1. *Assume $\{z'_i, x'_i\}_{i=1}^L$ is obtained by the ReLU network h_L with input z'_1, and $\{\delta_i^*\}_{i=0}^{L-1}$ is the optimal solution of $\mathcal{C}$. If $z'_1 = x + \delta_0^*$, and $x'_{ij} \in \mathcal{S}(\delta_{ij}^*)$ for all $i = 1, ..., L-1, j \in \mathcal{I}_i$, then $\{z'_i, x'_i\}_{i=1}^L$ is an optimal solution of $\mathcal{C}$. Here*

$$\mathcal{S}(\delta_{ij}^*) = \begin{cases} \{\underline{x}_{ij}, \overline{x}_{ij}\} & \text{if } \delta_{ij}^* = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}}, \\ \{0\} & \text{if } \delta_{ij}^* = 0. \end{cases}$$

We provide the proof of this simple proposition in the Appendix.

By Proposition 1, to evaluate the gap $p_{\mathcal{O}}^* - p_{\mathcal{C}}^*$, we only need to compute the sum of distances for all x'_{ij} corresponding to the unstable neurons in the set $\mathcal{S}(\delta_{ij}^*)$. Specifically, given the optimal solutions $\{\delta_i^*\}_{i=0}^{L-1}$ of $\mathcal{C}$, we pass $x + \delta_0^*$ through the equivalent ReLU network to obtain $x'_i = h_i(x + \delta_0^*)$, and compute the following as an indicator for the gap:

$$r(x, \delta_0^*, W, b) = \frac{1}{\sum_{i=1}^{L-1} |\mathcal{I}_i|} \sum_{i=1}^{L-1} \left(\sum_{\substack{j \in \mathcal{I}_i \\ \delta_{ij}^* = 0}} |x'_{ij}| + \sum_{\substack{j \in \mathcal{I}_i \\ \delta_{ij}^* \neq 0}} \min(|x'_{ij} - \underline{x}_{ij}|, |x'_{ij} - \overline{x}_{ij}|) \right), \quad (9)$$

where the first term corresponds to $\delta_{ij}^* = 0$ and the condition $x'_{ij} \in \{0\}$, and the second term corresponds to $\delta_{ij}^* = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij}-\underline{x}_{ij}}$ and the condition $x'_{ij} \in \{\underline{x}_{ij}, \overline{x}_{ij}\}$. To minimize the second term, the original ReLU network only needs to be optimized towards the nearest feasible optimal solution. It is easy to see from Proposition 1 that if $r(x, \delta_0^*, W, b) = 0$, then $p_{\mathcal{O}}^* = p_{\mathcal{C}}^*$.

Compared with $d(x, \delta_0^*, W, b)$, $r(x, \delta_0^*, W, b)$ puts more constraints on the parameters W, b , since it requires all unstable neurons of the ReLU network to match the feasible optimal solutions of $\mathcal{C}$, instead of only matching $p'_{\mathcal{O}}$ and $p_{\mathcal{C}}^*$. In this way, it provides stronger guidance towards a network whose optimal condition for $\mathcal{O}$ and $\mathcal{C}$ agree.

Algorithm 1 Computing the Bounds and Regularizers for ℓ_∞ Norm

Data: Clean images$\{x^{(j)}\}_{j=1}^m$, ReLU network with parameters W, b, and maximum perturbation ϵ.
Result: Margin Lower Bound $p_{\mathcal{C}}^*$, Regularizer $d(x, \delta_0^*, W, b)$ and $r(x, \delta_0^*, W, b)$.
Initialize $\mathcal{W}_{1:1} \leftarrow W_1$, $g_1(x) = W_1 x + b_1$, $\underline{x}_1 = g_1(x) - \epsilon \|\mathcal{W}_{1:1}\|_{1,row}$, $\bar{x}_1 = g_1(x) + \epsilon \|\mathcal{W}_{1:1}\|_{1,row}$
for *i=2,. . . ,L-1* **do**
  Compute D_{i-1} with Eq. 4, $\mathcal{W}_{i:1} \leftarrow W_i D_{i-1} \mathcal{W}_{i-1}, g_i(x) = W_i D_{i-1} g_{i-1}(x) + b_i$
  $\underline{x}_i \leftarrow g_i(x) - \epsilon \|\mathcal{W}_{i:1}\|_{1,row} - \sum_{i'=1}^{i-1} \sum_{j \in I_{i'}} \frac{\underline{x}_{i'j}\overline{x}_{i'j}}{\overline{x}_{i'j} - \underline{x}_{i'j}} \min((\mathcal{W}_{i:i'+1})_{:,j}, 0)$
  $\overline{x}_i \leftarrow g_i(x) + \epsilon \|\mathcal{W}_{i:1}\|_{1,row} + \sum_{i'=1}^{i-1} \sum_{j \in I_{i'}} \frac{\underline{x}_{i'j}\overline{x}_{i'j}}{\overline{x}_{i'j} - \underline{x}_{i'j}} \min(-(\mathcal{W}_{i:i'+1})_{:,j}, 0)$
end
Compute D_{L-1} with Eq. 4, $\mathcal{W}_{L:1} \leftarrow W_L D_{L-1} \mathcal{W}_{L-1}, g_L(x) = W_L D_{L-1} g_{L-1}(x) + b_L$
Compute $p_{\mathcal{C}}^*$ with Eq. 7, $\delta_0^* \leftarrow -\epsilon \|c_t^\top \mathcal{W}_{L:1}\|_{1,row}$
$x'_i \leftarrow h_i(x + \delta_0^*)$ **for** $i = 1$ to L
$d(x, \delta_0^*, W, b) \leftarrow c_t^\top x'_L - p_{\mathcal{C}}^*$
Compute $r(x, \delta_0^*, W, b)$ with Eq. 9

Dataset	Model	ϵ	λ	γ	Rob. Err	PGD Err	Std. Err
MNIST	2x100, Exact	0.1	0	0	14.85%	**10.9%**	**3.65%**
MNIST	2x100, Exact	0.1	2e-3	1	**13.32%**	**10.9%**	4.73%
MNIST	Small, Exact	0.1	0	0	4.47%	2.4%	1.19%
MNIST	Small, Exact	0.1	5e-3	5e-1	**3.65%**	**2.2%**	1.09%
MNIST	Best of (Dvijotham et al., 2018a)	0.1	-	-	4.44%	2.87%	1.20%
MNIST	Best of (Xiao et al., 2018)	0.1	-	-	4.40%	3.42%	**1.05%**
MNIST	Small	0.1	0	0	4.47%	**3.3%**	1.19%
MNIST	Small	0.1	0	5e-1	**4.32%**	3.4%	**1.51%**
MNIST	Large (Mirman et al., 2018)	0.1	-	-	3.4%	2.4%	1.0%
MNIST	2x100, Exact	0.3	0	0	61.39%	49.4%	33.16%
MNIST	2x100, Exact	0.3	5e-3	5e-3	**56.05%**	**44.3%**	**26.10%**
MNIST	Small, Exact	0.3	0	0	31.25%	15.0%	7.88%
MNIST	Small, Exact	0.3	5e-3	5e-1	**29.65%**	**13.7%**	**7.28%**
MNIST	Small	0.3	0	0	42.7%	26.0%	15.93%
MNIST	Small	0.3	2e-3	2e-1	**41.36%**	**24.0%**	**14.29%**
CIFAR10	Small	2/255	0	0	53.19%	48.0%	38.19%
CIFAR10	Small	2/255	5e-3	5e-1	**51.52%**	**47.0%**	**37.30%**
CIFAR10	Large	2/255	0	0	45.78%	**38.5%**	**29.42%**
CIFAR10	Large	2/255	5e-3	5e-1	**45.19%**	38.8%	29.76%
CIFAR10	Large (Mirman et al., 2018)	2/255	-	-	61.4%	55.6%	55.0%
CIFAR10	Small	8/255	0	0	75.45%	68.3%	62.79%
CIFAR10	Small	8/255	1e-3	1e-1	**74.70%**	**67.9%**	**62.50%**
CIFAR10	Large	8/255	0	0	74.04%	68.8%	**59.73%**
CIFAR10	Large	8/255	5e-3	5e-1	**73.74%**	**68.5%**	59.82%

Table 1: Results on MNIST, and CIFAR10 with small networks, large networks, and different coefficients of $d(x, \delta_0^*, W, b)$, $r(x, \delta_0^*, W, b)$. For all models not marked as "Exact", we have projected the input dimension of $\mathcal{W}_{i:1}$ to 50, the same as Wong et al. (2018).

4.4 Certified Robust Training in Practice

In practice, for classification problems with more than two classes, we will compute the lower bound of the margins w.r.t. multiple classes. Denote $\mathbf{p}_{\mathcal{C}}^*$ and $\mathbf{p}_{\mathcal{O}}^*$ as the concatenated vector of lower bounds of the relaxed problem and original problem for multiple classes, and d_t, r_t as the regularizers for the margins w.r.t. class t. Together with the regularizers, we optimize the following objective

$$\underset{W,b}{\text{minimize}}\ L_{CE}(-\mathbf{p}_{\mathcal{C}}^*, y) + \lambda \sum_t d_t(x, \delta_0^*, W, b) + \gamma \sum_t r_t(x, \delta_0^*, W, b), \tag{10}$$

where $L_{CE}(-\mathbf{p}_{\mathcal{C}}^*, y)$ is the cross entropy loss with label y, as adopted by many related works (Wong et al., 2018; Gowal et al., 2018), and we have implicitly abbreviated the inner maximization problem w.r.t. $\{\delta_i\}_{i=0}^{L-1}$ into $\mathbf{p}_{\mathcal{C}}^*$ and δ_0^*. More details for computing the intermediate and output bounds can be found in Algorithm 1, where we have used $\|\cdot\|_{1,row}$ to denote row-wise ℓ_1 norm, and $(\cdot)_{:,j}$ for taking the j-th column.

One major challenge of the convex relaxation approach is the high memory consumption. To compute the bounds $\underline{x}_i, \overline{x}_i$, we need to pass an identity matrix with the same number of diagonal entries as the total dimensions of the input images, which can make the batch size thousands of times larger than usual. To mitigate this, one can adopt the random projection from Wong et al. (2018), which projects identity matrices into lower dimensions as $\mathcal{W}_{i:1}R$ to estimate the norm of $\mathcal{W}_{i:1}$. Such projections add noise/variance to $\underline{x}_i, \overline{x}_i$, and the regularizers are affected as well.

5 Experiments

Models, Datasets and Hyper-parameters: We evaluate the proposed regularizer on two datasets (MNIST and CIFAR10) with two different ϵ each. We consider only ℓ_∞ adversaries. We experiment with a variety of different network structures, including a MLP (2x100) with two 100-neuron hidden layers as (Salman et al., 2019), two Conv Nets that are the same as (Wong et al., 2018), a family of

10 small conv nets and a family of 8 larger conv nets, all the same as (Zhang et al., 2019b). We give more details about the network structures and the hyper-parameters in Appendix.

Improved Training with Convex Relaxation Table 1 shows comparisons mainly with (Wong et al., 2018). All of our baseline implementations have improved compared with (Wong et al., 2018). When adding the proposed regularizers, the certified robust accuracy is further improved in all cases. We also provide results against a 100-step PGD adversary. Since both PGD errors and standard errors are reduced in most cases, the regularizer should have improved not only the certified upper bound, but also improved the actual robust error. The relative improvement on 2x100 with our regularizer (10.3%/8.7%) are comparable to the improvements (5.6%/10.0%) from (Salman et al., 2019), despite the fact that we start from a stronger baseline. Our results with Small are better than the best results of (Dvijotham et al., 2018a; Xiao et al., 2018) on MNIST with $\epsilon = 0.1$, though not as good as the best of (Mirman et al., 2018), which uses a larger model. When applying the same model on CIFAR10, we achieve better robust error than (Mirman et al., 2018). The relative improvements in certified robust error for $\epsilon = 0.1$ and 0.3 are 18%/3.4% for the small exact model on MNIST, compared with 0.03%/3.13% for the random projection counterparts. In the exact models, we have better estimates of $\underline{x}_i, \overline{x}_i$. These consistent improvements validate that our proposed regularizers improve the performance.

Dataset	$\epsilon(\ell_\infty)$	Model Family	Method	Verified Test Error (%)			Standard Test Error (%)		
				best	median	worst	best	median	worst
MNIST	0.3	Gowal et al. (2018)		8.05	-	-	1.66	-	-
		8 large models	CI Orig	7.46	8.47	8.57	1.48	1.52	1.99
			CI ReImp	7.99	8.38	8.97	1.40	1.69	2.19
			CI Reg	7.26	8.44	8.88	1.51	1.72	2.21

Table 2: Our results on the MNIST dataset, with CROWN-IBP. CI Orig are results copied from the paper, CI ReImp are results of our implementation of CROWN-IBP, and CI Reg is with regularizer r.

Method	error	model A	model B	model C	model D	model E	model F	model G	model H	model I	model J
Copied	std. (%)	$5.97 \pm .08$	3.20 ± 0	$6.78 \pm .1$	$3.70 \pm .1$	$3.85 \pm .2$	$3.10 \pm .1$	$4.20 \pm .3$	$2.85 \pm .05$	$3.67 \pm .08$	$2.35 \pm .09$
	verified (%)	$15.4 \pm .08$	$10.6 \pm .06$	$16.1 \pm .3$	$11.3 \pm .1$	$11.7 \pm .2$	$9.96 \pm .09$	$12.2 \pm .6$	$9.90 \pm .2$	$11.2 \pm .09$	$9.21 \pm .3$
Baseline	std. (%)	$5.65 \pm .04$	$3.23 \pm .3$	$4.70 \pm .4$	$2.94 \pm .05$	$6.39 \pm .3$	$2.89 \pm .05$	$4.11 \pm .3$	$2.55 \pm .1$	$3.30 \pm .4$	$2.56 \pm .1$
	verified (%)	$14.70 \pm .07$	$10.65 \pm .3$	$13.78 \pm 1.$	$9.89 \pm .3$	$15.26 \pm .6$	$9.54 \pm .1$	$12.06 \pm .9$	$8.92 \pm .2$	$10.81 \pm .8$	$9.67 \pm .4$
With r	std. (%)	$5.80 \pm .04$	$3.16 \pm .06$	$5.16 \pm .3$	$3.06 \pm .05$	$6.15 \pm .2$	$2.95 \pm .05$	$3.83 \pm .2$	$2.57 \pm .1$	$3.29 \pm .04$	$2.73 \pm .4$
	verified (%)	$14.54 \pm .07$	$10.46 \pm .08$	$13.37 \pm .3$	$9.91 \pm .3$	$14.43 \pm .3$	$9.48 \pm .2$	$11.01 \pm .5$	$8.83 \pm .3$	$10.18 \pm .5$	$9.49 \pm .4$

Table 3: Mean and standard deviation of the family of 10small models on MNIST with $\epsilon = 0.3$. Baseline is CORWN-IBP with epoch=140 and lr_decay_step=20. Like in CROWN-IBP, we run each model 5 times to compute the mean and standard deviation. "Copied" are results from (Zhang et al., 2019b).

Improved Training with CROWN-IBP Zhang et al. (2019b) propose to use a linear combination of the convex relaxation bound (CROWN) and the IBP bound in the early stage to stablize the training of IBP. We find applying regularizer r together with CROWN and gradually phase it out improves the final solution. Table 3 shows the mean and variance of the results with smaller models, demonstrating consistent improvements of our model, while Table 2 gives the best, median and worst case results with the large models. Note all the networks are significantly smaller than (Gowal et al., 2018), and our batch size and number of epochs (at most 256 and 140) are also much smaller than (Gowal et al., 2018) (1600 and 3200). Yet, we achieved better results.

6 CONCLUSIONS

We propose two regularizers that lead to tighter LP relaxation bounds for certifiable robustness. Extensive experiments validate that the regularizers improve robust accuracy over non-regularized baselines. This work is a step towards closing the gap between certified and empirical robustness. Future directions include methods to improve computational efficiency for LP relaxations (and certified methods in general), and better ways to leverage random projections for acceleration.

REFERENCES

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14. ACM, 2017.

Nicholas Carlini, Guy Katz, Clark Barrett, and David L Dill. Provably minimally-distorted adversarial examples. *arXiv preprint arXiv:1709.10207*, 2017.

Chih-Hong Cheng, Georg Nührenberg, and Harald Ruess. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pp. 251–268. Springer, 2017.

Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. Output range analysis for deep feedforward neural networks. In *NASA Formal Methods Symposium*, pp. 121–138. Springer, 2018.

Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*, 2018a.

Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. *arXiv preprint arXiv:1803.06567*, 104, 2018b.

Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pp. 269–286. Springer, 2017.

Matteo Fischetti and Jason Jo. Deep neural networks as 0-1 mixed integer linear programs: A feasibility study. *arXiv preprint arXiv:1712.06174*, 2017.

Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pp. 2266–2276, 2017.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Uiwon Hwang, Jaewoo Park, Hyemi Jang, Sungroh Yoon, and Nam Ik Cho. Puvae: A variational autoencoder to purify adversarial examples. *arXiv preprint arXiv:1903.00585*, 2019.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pp. 97–117. Springer, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alessio Lomuscio and Lalit Maganti. An approach to reachability analysis for feed-forward relu neural networks. *arXiv preprint arXiv:1706.07351*, 2017.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pp. 3575–3583, 2018.

Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robust verification of neural networks. *NeurIPS*, 2019.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Karsten Scheibler, Leonore Winterer, Ralf Wimmer, and Bernd Becker. Towards verification of artificial neural networks. In *MBMV*, pp. 30–40, 2015.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

Shawn Shan, Emily Willson, Bolun Wang, Bo Li, Haitao Zheng, and Ben Y. Zhao. Gotta catch 'em all: Using concealed trapdoors to detect adversarial attacks on neural networks, 2019.

Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*, pp. 10802–10813, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. *arXiv preprint arXiv:1802.05666*, 2018.

Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. *ICML*, 2018.

Eric Wong and J Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*, 2017.

Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems*, pp. 8400–8409, 2018.

Kai Y Xiao, Vincent Tjeng, Nur Muhammad Shafiullah, and Aleksander Madry. Training for faster adversarial robustness verification via inducing relu stability. *arXiv preprint arXiv:1809.03008*, 2018.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Painless adversarial training using maximal principle. *arXiv preprint arXiv:1905.00877*, 2019a.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems*, pp. 4939–4948, 2018.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019b.

A PROOF OF PROPOSITION 1

Proposition 2. *Assume $\{z'_i, x'_i\}_{i=1}^{L}$ is obtained by the ReLU network h_L when input is z'_1, and $\{\delta_i^*\}_{i=0}^{L-1}$ is the optimal solution of $\mathcal{C}$. If $z'_1 = x + \delta_0^*$, and $x'_{ij} \in \mathcal{S}(\delta_{ij}^*)$ for all $i = 1, ..., L-1, j \in \mathcal{I}_i$, then $\{z'_i, x'_i\}_{i=1}^{L}$ is an optimal solution of $\mathcal{C}$. Here*

$$\mathcal{S}(\delta_{ij}^*) = \begin{cases} \{\underline{x}_{ij}, \overline{x}_{ij}\} & \text{if } \delta_{ij}^* = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}, \\ \{0\} & \text{if } \delta_{ij}^* = 0. \end{cases}$$

Proof. Here we define $x_i^* = W_i z_i + b_i$ for $i = 1, ..., L$, $z_{i+1}^* = D_i x_i^* + \delta_i^*$ for $i = 1, ..., L-2$, and $z_1^* = x + \delta_0^*$. By definition, $\{x_i^*, z_i^*\}_{i=1}^{L}$ is an optimal solution of $\mathcal{C}$. Also, since $z_1^* = z'_1$, we have $x'_1 = W_1 z_1^* + b_1 = x_1^*$.

For $j \in \mathcal{I}_1^+$, $D_{1,j} = 1$, and $x'_1 = x_1^* \geq \underline{x}_{ij} > 0$, so $z_{2,j}^* = D_{1,j} x_{1,j}^* = x'_{1,j} = \max(x'_{1,j}, 0) = z'_{2,j}$.

For $j \in \mathcal{I}_1^-$, $D_{1,j} = 0$, and $x'_1 = x_1^* \leq \overline{x}_{ij} < 0$, so $z_{2,j}^* = D_{1,j} x_{1,j}^* = 0 = \max(x'_{1,j}, 0) = z'_{2,j}$.

For $j \in \mathcal{I}_1$, $D_{1,j} = \frac{\overline{x}_{1,j}}{\overline{x}_{1,j} - \underline{x}_{1,j}}$. If $\delta_{1,j}^* = 0$ and $x'_{1,j} = 0$ as assumed in the conditions, then we know $x_{1,j}^* = 0$. So $z_{2,j}^* = D_{1,j} x_{1,j}^* = 0$, $z'_{2,j} = \max(x'_{1,j}, 0) = 0$, the equality still holds. If $\delta_{1,j}^* = -\frac{\overline{x}_{ij}\underline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}$, and $x'_{1,j} \in \{\underline{x}_{ij}, \overline{x}_{ij}\}$ as assumed: if $x'_{1,j} = \underline{x}_{1,j}$, then $x'_{1,j} < 0$, so $z'_{2,j} = 0$, and $z_{2,j}^* = \frac{\overline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}(\underline{x}_{1,j} - \underline{x}_{1,j}) = 0 = z'_{2,j}$; if $x'_{1,j} = \overline{x}_{1,j}$, then $x'_{1,j} > 0$, $z'_{2,j} = x'_{1,j}$ and $z_{2,j}^* = \frac{\overline{x}_{ij}}{\overline{x}_{ij} - \underline{x}_{ij}}(\overline{x}_{1,j} - \underline{x}_{1,j}) = \overline{x}_{1,j} = z'_{2,j}$.

Now we have proved $z'_2 = z_2^*$. Starting from this layer, using the same argument as above, we can prove $z'_3 = z_3^*$,...,$z'_{L-1} = z_{L-1}^*$. As a result, $x'_L = x_L^*$ and $c_t^T x'_L = c_t^T x_L^* = p_{\mathcal{C}}^*$ □

B EXPERIMENTAL DETAILS

Experiments for Table 1 The small convnet has two convolutional layers of 16, 32 output channels each and two FC layers with 100 hidden neurons. The large net has four Conv layers with 32, 32, 64 and 64 output channels each, plus three FC layers of 512 neurons. For all experiments, we are using Adam (Kingma & Ba, 2014) with a learning rate of $1e-3$. Like (Wong et al., 2018), we train the models for 80 epochs, where in the first 20 epochs the learning rate is fixed but the ϵ increases from 0.01/0.001 to its maximum value for MNIST/CIFAR10, and in the following epochs, we reduce learning rate by half every 10 epochs. We do not use weight decay. We also adopt a warm-up schedule in all experiments, where λ, γ increases form 0 to the preset values in the first 20 epochs, due to the noisy estimation from the random projections. Our implementation is based on the code released by (Wong et al., 2018), so when $\lambda = \gamma = 0$, we obtain the same results as as (Wong et al., 2018).

Experiments for Other Tables We use the same hyper-parameters as (Zhang et al., 2019b), except for "lr_decay_step" and "epochs", which are set to 20 and 140 for lower variance.

C ABLATION STUDIES OF THE TWO REGULARIZERS

In this section, we give the detailed results with either λ or γ set to 0, i.e., we use only one regularizer in each experiment, in order to compare the effectiveness of the two regularizers. All the results are with the small model on CIFAR10 with $\epsilon = 2/255$. The best results are achieved with $r(x, \delta_0^*, W, b)$. We reasoned in 4.2 that $d(x, \delta_0^*, W, b)$ may not perform well when random projection is adopted. As shown in the supplementary, the best robust error achieved under the same setting when fixing $\gamma = 0$ is higher than when fixing $\lambda = 0$, which means $r(x, \delta_0^*, W, b)$ is more resistant to the noise introduced by random projection. Still, random projections offer a huge efficiency boost when they are used. How to improve the bounds while maintaining efficiency is an important future work.

λ	γ	**Robust Err (%)**	**Std. Err (%)**	λ	γ	**Robust Err (%)**	**Std. Err (%)**
1e-5	0	52.90	37.61	0	1e-3	52.56	37.45
5e-5	0	53.09	37.77	0	5e-3	53.38	38.19
1e-4	0	52.48	37.37	0	1e-2	52.60	37.73
5e-4	0	52.85	37.13	0	2.5e-2	52.70	37.78
1e-3	0	52.61	37.96	0	5e-2	53.13	38.36
2.5e-3	0	53.10	38.24	0	1e-1	52.72	38.22
5e-3	0	52.76	38.15	0	2.5e-1	52.90	38.04
1e-2	0	53.14	38.58	0	5e-1	52.39	37.48
5e-2	0	52.82	39.89	0	1	**52.27**	**38.07**
1e-1	0	53.94	41.59	0	2	53.10	38.64
5e-1	0	56.39	48.06				
1	0	59.23	52.51				